IMPLEMENTATION OF MODIFIED NLMS ALGORITHM FOR ACOUSTIC ECHO CANCELLATION

M. Medvecký [+] *and S. Herrera-Garcia* [++]

[+] Dept. of Telecommunications, Slovak University of Technology in Bratislava, Bratislava, Slovak Republic
medvecky@elf.stuba.sk

[++] Communication Section of Electrical Engineering Dept., Reserch Center - Cinvestav, Mexico City, Mexico
sherrera@mvax1.red.cinvestav.mx

ABSTRACT

The implementation of modified version of Normalised Least-Mean-Square algorithm (M-NLMS) will be presented. The M-NLMS algorithm was developed for efficient weight adaptation of acoustic echo canceller based on modified finite impulse response (M-FIR) filter [1]. The M-FIR filter is especially suitable for hardware realisation of acoustic echo canceller. The M-NLMS algorithm enables maximum exploitation of advantages of the M-FIR filter. The proposed adaptive algorithm modification and implementation principles can be applied to other LMS based algorithms as well.

The comparison of acoustic echo cancellers with adaptive FIR filter adapted by NLMS algorithm and adaptive M-FIR filter adapted by M-NLMS algorithm are presented.

1. INTRODUCTION

The main problem of acoustic echo cancellation is the precise identification of the acoustic impulse response. The actual structure of the echo path is usually modelled by a FIR filter. Such a filter has the advantages of guaranteed stability during adaptation, unimodal means square error (MSE) surface and the principle of FIR filter response computation is similar to the acoustic echo origination process. The FIR filter may require up to thousands adaptive weights to correctly identify the acoustic impulse response. Such a number of taps result in a large arithmetic complexity that is a crucial problem for real time implementations like handsfree telephones or videoconference systems. The solution is a hardware realisation of the acoustic echo canceller that enables a parallel processing implementation.

Unfortunately, the hardware realisation brings a problem with a computation precision. Arithmetic precision has a profound impact on realisation complexity. The hardware realisations of the acoustic echo canceller by VLSI or ASIC circuits use from the reason of simpler realisation fixed point arithmetic that yields to poorer performance than floating point arithmetic. Furthermore, the room impulse response has a very specific shape: a short delay followed by an exponentially decaying tail (see Fig.1). The modelling of such a characteristic by a transversal filter with fixed point arithmetic leads due to roundoff error to gradual degradation of a filter weights accuracy in direction toward the end of the filter. This yields to a poorer performance of the acoustic echo canceller.

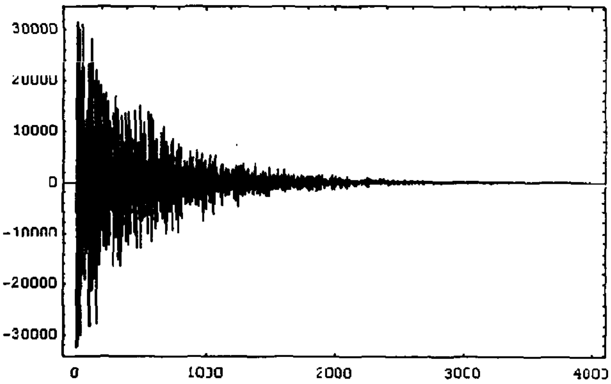

Fig. 1 Impulse response of a teleconference room

This problem can be solved by the partitioning of the impulse response in time or in frequency domain for each partition [2]. Then, the filter weights may have a different gain in each block. Since the room impulse response changes - the partitions and gains should be changed as well. To solve this problem a modified structure of the FIR (M-FIR) filter and the modified version of a NLMS (M-NLMS) algorithm has been derived.

2. MODIFIED FIR FILTER

2.1 Filter derivation

Consider an impulse response of communication environment that is typical for hands-free telephones or videoconference systems (see Fig.1).

When we use an adaptive transversal filter to model such an impulse response then the output y_k of the transversal filter can be obtained as a sum of weighted contributions from the last N samples of the input signal x.
It can be written as follow

$$y_k = \sum_{i=0}^{N-1} y_{ik} = \sum_{i=0}^{N-1} x_{k-i} w_{ik} \qquad (1)$$

where w_{ik} are weight of the adaptive filter in time k. If d_k is a sample of a desired signal in time k than the error e_k is given as

$$\begin{aligned} e_k &= d_k - y_k = d_k - (y_{0k} + y_{1k} + \ldots + y_{(N-1)k}) \\ &= (d_k - y_{0k}) - y_{1k} - \ldots - y_{(N-1)k} \end{aligned} \qquad (2)$$

Applying the following substitutions

$$d_{0k} = d_k \qquad (3)$$

$$e_{ik} = d_{ik} - y_{ik} \qquad (4)$$

$$d_{(i+1)k} = e_{ik} \qquad (5)$$

we obtain a new topology of an adaptive filter. The block diagram of the Modified FIR filter is shown in Fig.2.

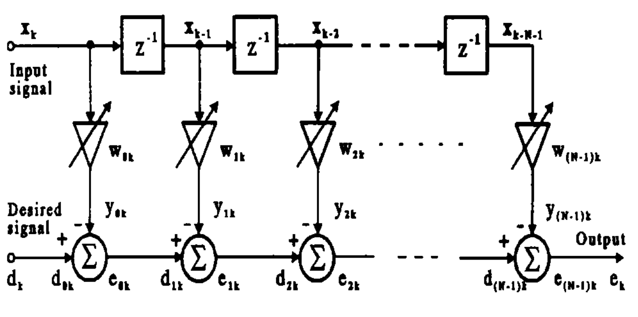

Fig. 2 Modified FIR filter

2.2 Implementation

As can be seen, the values of the error e_{ik} are decreasing with raising i. Since the values of the weight w_{ik} are decreasing as well we can increase the computation accuracy by multiplying of the e_{ik} and w_{ik} when their values fall under the specific level (threshold). The multiplying can be made adaptive. The choice of the multiplier coefficients as 2^n enables to realise the multiplication very simple by rotation (bit shifting). The hardware realisation of such an adaptive filter can have the computation complexity equal to the complexity of the classical FIR filter. More details about the M-FIR filter implementations and internal data representations can be found in [3].

3. MODIFIED NLMS ALGORITHM

3.1 Derivation

Consider a situation that the real signal and filter with coefficient represented by real numbers should be realised by the system with fixed point arithmetic. In this case, scaling of data and coefficients to best fit within the dynamic range of the used system is required. In practical terms, scaling is reduced to selection of crest factor appropriate for the signal characteristics and the precision used in storage and arithmetic. In that case, the internal scaling can be realised by the normalisation to the value δ that is a signal power level rounded to the nearest 2^n value. The variable δ represents a basic shift in internal number representation.

Considering the above implementation the convergence factor μ for M-NLMS algorithm can be express as

$$\mu_k = \alpha^* / (\gamma^* + \mathbf{X}_k^T \mathbf{X}_k) \qquad (6)$$

where α^* and γ^* are coefficients α and γ from NLMS algorithm shifted by the value δ. The term $\mathbf{X}_k^T \mathbf{X}_k$ in equation (6) can be obtain for time k by iteration

$$\mathbf{X}^T_k \mathbf{X}_k = \mathbf{X}^T_{k-1} \mathbf{X}_{k-1} - x^2_{k-N-1} + x^2_k \qquad (7)$$

and in our case

$$\mathbf{X}^T_k \mathbf{X}_k = \mathbf{X}^T_{k-1} \mathbf{X}_{k-1} - x^2_{k-N-1} / \delta + x^2_k / \delta \qquad (8)$$

The new values of the filter weight can be obtained as follow. Let we define

$$w^*_{i(k+1)} = (w_{ik}\, \delta) / \delta_{w_k} + ((\mu_k e_k x_{k-i}) / \delta) / \delta_e \qquad (9)$$

and

$$w^{**}_{i(k+1)} = (w^*_{i(k+1)}\, \delta_{w_{k+1}}) / \delta \qquad (10)$$

where δ_{w_k} is an additional shift of the weight w_i in time k compare to the basic shift δ and $\delta_{w_{k+1}}$ is an additional shift of the filter weight w_i in time k+1 compare to the basic shift δ. The e_k is a final error (see Fig.2.) obtained in time k and δ_e is additional shift of this error. The brackets in equations (9) and (10) define the sequence of arithmetic operations execution that guarantee that the entire term can be retained without prematurely truncating its precision and without resorting to extended-width of internal buses.
Next we can define following six conditions

$$C_1 = 1 \quad \text{if } |w^{**}_{i(k+1)}| > \Psi^M_w \qquad (11a)$$

$$C_2 = 1 \quad \text{if } |w^{**}_{i(k+1)}| > \Psi^H_w \qquad (11b)$$

$$C_3 = 1 \quad \text{if } \delta_{w_{k+1}} > 1 \qquad (11c)$$

$$C_4 = 1 \quad \text{if } |w^{**}_{i(k+1)}| < \Psi^L_w \qquad (11d)$$

$$C_5 = 1 \quad \text{if } |w^{**}_{i(k+1)}| <> 0 \qquad (11e)$$

$$C_6 = 1 \quad \text{if } \delta_{w_{k+1}} < \delta^M_w \qquad (11f)$$

where δ^M_w is a maximum acceptable additional shift of the weight w_i. The threshold Ψ^M_w is a maximum value of w_i. The thresholds Ψ^H_w and Ψ^L_w define high and low threshold for the weight w_i. When the value w_i exceed them it can be increased/decreased (shifted up/down) by the unit step δ_1.

Now we can define four logical functions

$$f_1 = C_1 \quad (12a)$$

$$f_2 = C_2 \wedge C_3 \wedge \overline{f_1} \quad (12b)$$

$$f_3 = C_4 \wedge C_5 \wedge C_6 \wedge \overline{f_2} \quad (12c)$$

$$f_4 = \overline{f_3} \quad (12d)$$

If the value of the function $f_1 = 1$ (True), than

$$\delta_{w_{k+1}} = 1 \quad (13)$$

$$w_{i(k+1)} = w^*_{i(k+1)} / \delta \quad (14)$$

else if function $f_2 = 1$ (True), than

$$\delta_{w_{k+1}} = \delta_{w_{k+1}} / \delta_1 \quad (15)$$

$$w_{i(k+1)} = w^{**}_{i(k+1)} / \delta_1 \quad (16)$$

if not, than if $f_3 = 1$ (True)

$$\delta_{w_{k+1}} = \delta_{w_{k+1}} \delta_1 \quad (17)$$

$$w_{i(k+1)} = w^{**}_{i(k+1)} \delta_1 \quad (18)$$

and if the function $f_4 = 1$ (True), than

$$w_{i(k+1)} = w^{**}_{i(k+1)} \quad (19)$$

and $\delta_{w_{k+1}}$ does not change. The value of δ_w can be computed from two one bit (Boolean) variables δ^U and δ^D which indicate the shift of δ_w in stage i against the shift in stage i-1. This approach enables to save a storage space and is implemented in M-FIR filter. The new values of δ^U and δ^D are set-up as follow

$$\delta^U_i = f_1 \wedge f_3 \quad (20)$$

$$\delta^D_i = f_1 \wedge f_2 \quad (21)$$

3.2 Implementation

There are three possible realisations of adaptive echo canceller. They are:

- implementation on universal processor,
- implementation on DSP,
- hardware realisation.

and combination of these three methods.

The M-NLMS algorithm contains following operations:

- inversions,
- absolute values,
- arithmetic operations,
- logical operations,
- comparisons and conditions,
- cycles.

The hardware implementation of the M-NLMS algorithm enable the following realisation of these operations.

The conditions (11) can be solved simply by comparators. The function (12), (20) and (21) can be generated by logic elements (AND) or by look-up tables.

When the thresholds and the unit step δ_1 are set as numbers 2^n than the multiplication and division in equations (8 - 18) can be realised by multiplexer, demultiplexer or simply by bit shifting. The divisions by the value δ in equations (8), (9), (10) and (14) that represent the constant bit shift can be realised very simply by hardwiring and therefore it have no impact on computation complexity.

The conditions (11) and functions (12) can be evaluated parallel and very fast. For example, today available Field Programmable Gate Arrays (FPGA) Xilinx [4] have complexity more than 100000 equivalent gates and they can evaluate any function up to nine variables in 2 ns [5].

As can be seen the hardware realisation of the M-NLMS algorithm can decrease the computation complexity to the complexity of the classical NLMS algorithm.

4. SIMULATION RESULTS

To verify the better performance of the Modified NLMS algorithm for acoustic echo cancellation the following simulations were carried out. In the simulation an impulse response of a real teleconference room with length 4000 samples has been suppressed. The two adaptive filters with 4000 coefficients have been used. The FIR filter has been adapted by NLMS algorithm and M-FIR filter has been adapted by M-NLMS algorithm. The filter parameters were the same. In both cases the real data were scaled and internally represented by 16 bit integer with 11 bit for decimal part (the basic shift $\delta = 2048$). The unit step for M-NLMS algorithm was $\delta_1 = 2$. The measuring signal was the white Gaussian noise. Total 50000 iterations have been made for each experiment. The convergence characteristics of NLMS and M-NLMS algorithms are shown in Fig. 3. As can be seen, while NLMS algorithm reaches only 30dB echo suppression, M-NLMS algorithm overcomes a 40 dB level defined by ITU-T.

Dependencies of the level of acoustic echo cancellation on the beginning value of normalised adaptation coefficient α for NLMS and M-NLMS algorithm are shown in Fig. 4. As can be seen, the higher level of acoustic echo cancellation is reached by M-FIR filter adapted by M-NLMS algorithm. Furthermore, while NLMS algorithm reaches the maximum level of acoustic echo cancellation for higher values of normalised convergence parameter $\alpha \approx 2$ and faster convergence for $\alpha \approx 1$ then M-NLMS algorithm reaches the maximum level of acoustic echo cancellation and the fastest convergence for the same value $\alpha \approx 1$. (Note: parameter α can be in range $0 < \alpha < 2$) Therefore, the choice $\alpha = 1$ can decrease the computation complexity of M-NLMS algorithm.

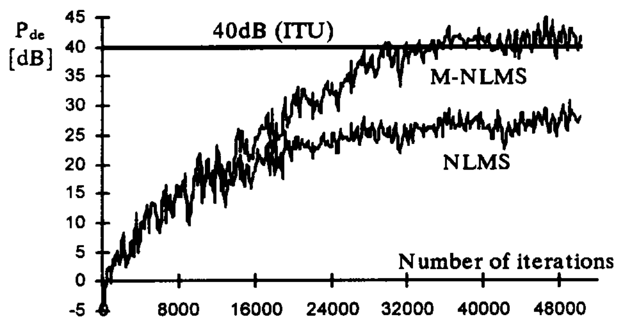

Fig. 3. Comparison of convergence characteristics of echo cancellers adapted by NLMS and M-NLMS algorithms

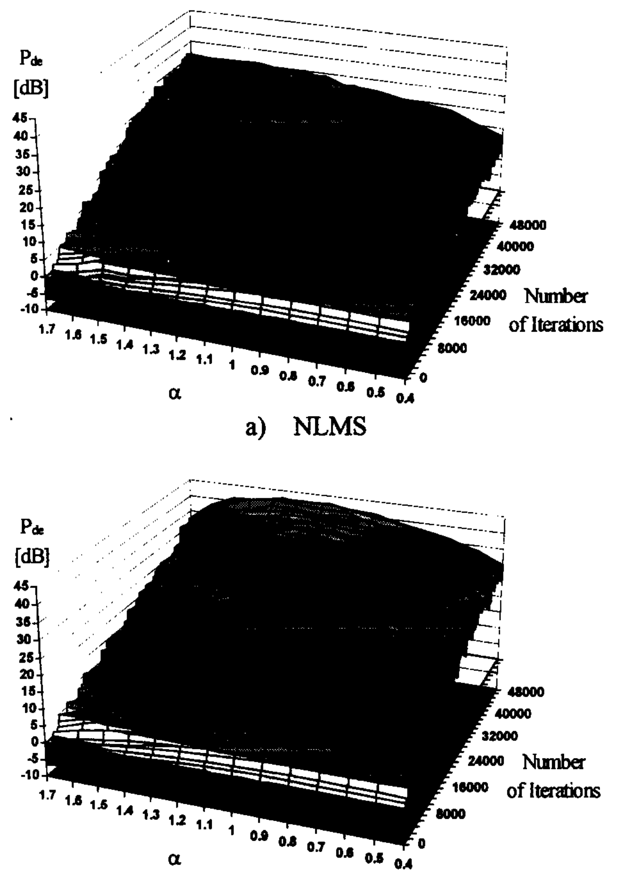

Fig. 4. Dependence of the level of acoustic echo cancellation on the normalised adaptation coefficient α

5. CONCLUSIONS

In this paper we have presented an implementation of modified version of NLMS algorithm (M-NLMS) and modified structure of FIR filter on adaptive acoustic echo cancellation. It was shown that the adaptive M-FIR filter and M-NLMS adaptive algorithm can achieve better performance for acoustic echo cancellation then the FIR filter and NLMS algorithm with the same parameters.

The implementation of weight shifting algorithm enables better exploitation of the dynamic range given by the number of bits for data representation. The effect of adaptive weight shifting is similar to the floating point arithmetic, but its hardware implementation is much simpler. As we have shown, the hardware realisation of proposed algorithm and filter enables achieve the same computation complexity as classic FIR filter and NLMS algorithm.

The proposed adaptive algorithm modification and implementation can be applied to other LMS based algorithms as well.

REFERENCES

[1] MEDVECKÝ. M.: Modified FIR Filter for Acoustic Echo Cancellation, In: 3rd International Workshop on Image and Signal Processing, Advances in Computational Intelligence, November 4-7, 1996, Manchester, UK

[2] BORRALLO, J. M. P., OTERO, M. G.: "On the Implementation of a Partitioned Block Frequency Domain Adaptive Filter (PBFDAF) for Long Acoustic Echo Cancellation". Signal Processing, Vol. 27, 1992, pp. 301-315.

[3] MEDVECKÝ.M.: "Hardware Realisation of Acoustic Echo Canceller in Hands-free Telephones". In.: 3rd International Conference TELEKOMUNIKACIE' 97, Bratislava, 4.-5.6.1997, 1997, pp.160-164. (in Slovak)

[4] The Programmable Logic Data Book, Xilinx, 1994.

[5] http://www.xilinx.com